FIG. **10** shows the preferred general flow of the Runstep program. Runstep program **860** is run in a loop with a Runstep batch file **870**. Runstep program **860** reads and interprets a step in a step sequence and writes the command to be run from the step sequence into batch file **870**. Batch file **870** is then run, executing the step upon target computer system **160**. Upon completion of a step, control is returned from the batch file to Runstep program **860** which then reads and interprets the next line of the step sequence.

FIG. **11** shows a more detailed flow of the Runstep program. As illustrated in module **900**, the Runstep program first checks to see if a file named Re_Run.bat exists. A Re_Run.bat file is created before any command is executed from a step sequence and is removed after successful completion of the command. The existence of Re_Run.bat indicates to the Runstep program in module **900** that the last command run was not successfully completed. Thus Re_Run.bat functions as a start of execution indication. If Re_Run.bat does exist, an operator is asked in module **904** whether or not the software installation and testing process should continue or whether the operator prefers instead to perform troubleshooting. If an operator chooses to continue, then control passes to execute module **928** where the Runstep.bat file is reexecuted. (This condition is the default option if neither option is affirmatively chosen.) If the troubleshooting option is chosen, then troubleshooting is performed as is well known in the art.

If Re_Run.bat does not exist, then the Runstep program determines that the last command was completed correctly, and control is passed to module **910**, where a line of the step sequence, preferably contained in a step file, is read. The Runstep program reads the line and determines if there is a beginning or ending timestamp in module **912**. If there is a beginning or ending timestamp, then the Runstep program determines, in module **914**, whether there is only a beginning timestamp for the line that the Runstep program is reading. If there is only a beginning timestamp, then the Runstep program assumes in module **916** that a software installation or testing step has just been finished and fills in an ending timestamp in module **918**. After filling in the ending timestamp, control is returned to module **900**.

If there is more than just a beginning timestamp for the line that the Runstep program is reading, the Runstep program determines in module **906** whether there is both a beginning and an ending timestamp. If so, then the Runstep program assumes in module **908** that the step has been executed and control is returned to module **900**. If the Runstep program encounters no beginning or ending timestamp in module **912**, then the Runstep program fills in the beginning timestamp in module **920** and prepares to run the step on the line of the step sequence that the Runstep program is reading.

In module **922**, the Runstep program determines if the command to be run is stored on a local drive (the step file controls which drive in the system is the local drive). The local drive may be, e.g., the step disk, a hard drive of the target system, a RAM drive of the target system, or a network drive. If the command is not located on the local drive, then the Runstep program assumes that the test to be run is contained on a file server somewhere on a network. The Runstep program determines in module **932** whether the Runstep program is already connected to that network. If not, the Runstep program, in module **936**, embeds a command into Runstep.bat to login to the network. Therefore a network connection is made before Runstep.bat executes the step on target computer system **160** over network connection **180**.

Following module **936**, control is passed to module **926**. If the Runstep program is already logged into the network the Runstep program, during module **934**, removes commands from Runstep.bat to login to the network, for an additional login step is unnecessary if a network connection already exists. Control is then passed to module **926**. If the step to be run happens to be on step disk **150**, the Runstep program need not log into the network. Thus, in module **924**, the Runstep program removes commands from Runstep.bat to login to the network. Control is then passed to module **926**. In module **926**, the Runstep program embeds the proper command to be run into Runstep.bat and into Re_Run.bat. The command so embedded is taken from the step sequence, preferably contained in the step file. In module **928**, the step is executed by running Runstep.bat and, if executed successfully, Re_Run.bat is deleted. If the step is not executed successfully, then the Re_Run.bat file is not deleted and control transfers to failure state **929**. Control is then returned to module **900** so that another line may be read from the step sequence. This process continues until all the software installation and testing steps are completed.

Upon execution of the step sequence, the target system is tested and software is installed. In the embodiment of FIG. **1**, a select number of tests maybe run directly from step disk **150**, but the majority of tests are run from file server **190** over network connection **180**. Running tests from file server **190** advantageously eliminates limitations imposed by the storage capacities of floppy disks such as step disk **150**.

In the embodiment of FIG. **2**, the steps are run from file server **190** over network connection **180**. A floppy disk, here boot disk **220**, is needed only to boot target computer system **160**. Such a system advantageously simplifies the software installation and testing process.

Turning once again to FIGS. **1** and **2**, arrow **210** depicts that results from the software installation and testing may be logged back to either file server **190** or to file server **202**. The results preferably include whether all the steps were completed successfully and what types of failures (if any) were encountered. Logging the results might include simply saving or writing a modified version of the step file following the execution of the step sequence, for as discussed above, the step file is timestamped by the Runstep program. Such a system advantageously allows for improved troubleshooting capabilities during computer system manufacturing.

While particular embodiments of the present invention have been shown and described, it will be obvious to those skilled in the art that changes and modifications may be made without departing from this invention in its broader aspects and, therefore, the appended claims are to encompass within their scope all such changes and modifications as fall within the true spirit and scope of this invention.

What is claimed is:

1. A database for use by a system database in the manufacturing of a build-to-order computer system comprising:

a step table containing a set of software installation and testing steps shared among different components of substantially all computer systems being manufactured, wherein a prescribed software installation or testing step is executed by the system database during the manufacturing of the build-to-order computer system to facilitate a corresponding software installation or testing for the build-to-order computer system, the step table including an aftercode attribute identifying whether a halt or reboot is required after a corresponding step is executed; and

a component table coupled to the step table, the component table containing a set of substantially all possible